FlowLearn: Evaluating Large Vision-Language Models on Flowchart Understanding

Huitong Pan[a,*], **Qi Zhang**[a], **Cornelia Caragea**[b], **Eduard Dragut**[a] **and Longin Jan Latecki**[a]

[a]Dept. of Computer and Information Sciences, Temple University, Philadelphia
[b]Dept. of Computer Science, University of Illinois Chicago

Abstract. Flowcharts are graphical tools for representing complex concepts in concise visual representations. This paper introduces the FlowLearn dataset, a resource tailored to enhance the understanding of flowcharts. FlowLearn contains complex scientific flowcharts and simulated flowcharts. The scientific subset contains 3,858 flowcharts sourced from scientific literature and the simulated subset contains 10,000 flowcharts created using a customizable script. The dataset is enriched with annotations for visual components, OCR, Mermaid code representation, and VQA question-answer pairs. Despite the proven capabilities of Large Vision-Language Models (LVLMs) in various visual understanding tasks, their effectiveness in decoding flowcharts—a crucial element of scientific communication—has yet to be thoroughly investigated. The FlowLearn test set is crafted to assess the performance of LVLMs in flowchart comprehension. Our study thoroughly evaluates state-of-the-art LVLMs, identifying existing limitations and establishing a foundation for future enhancements in this relatively underexplored domain. For instance, in tasks involving simulated flowcharts, GPT-4V achieved the highest accuracy (58%) in counting the number of nodes, while Claude recorded the highest accuracy (83%) in OCR tasks. Notably, no single model excels in all tasks within the FlowLearn framework, highlighting significant opportunities for further development.

1 Introduction

Flowcharts are visual tools that simplify complex processes and concepts across various domains, condensing intricate information into concise visual representations that enhance both comprehension and communication. In this paper, a flowchart is defined as a diagram that outlines a sequence of operations using standardized symbols like rectangles for steps and arrows to indicate process flow, as demonstrated in Figure 1.

Flowchart comprehension, particularly in the domain of computer vision and LVLMs, remains underexplored despite their extensive application. Resources like ACL-Fig [13] that include scientific flowcharts are limited and often provide only basic figure captions and sparse inline reference annotations. Flowcharts' complex nature—requiring text recognition, identification of various visual elements (e.g., boxes, nodes, symbols), and understanding of node connections—demands more comprehensive annotations for effective evaluation, emphasizing the need for specialized resources.

Further underscoring the inadequacy of current resources, our preliminary analysis of 208 flowcharts from ACL-Fig using Gemini-Pro-Vision [9] yielded a low BLEU score of 0.006 (refer to supplementary material for details). However, this score doesn't fully represent the model's comprehension capabilities. The captions in ACL-Fig, sometimes as brief as 'Figure 2: Alignment learning algorithm', do not provide robust ground truths for meaningful evaluation. With a median caption length of just nine words, ACL-Fig is inadequate for evaluating flowchart understanding.

Addressing these gaps, we introduce the FlowLearn Dataset[1], which includes both scientific and simulated flowcharts. The scientific subset features 3,858 flowcharts sourced from scientific literature, annotated with captions (median length of 25 words) and in-figure text. The simulated subset consists of 10,000 flowcharts generated from Mermaid code. This simulated subset enhances the dataset by providing detailed annotations of visual components, thereby enabling quantitative evaluations of component-specific tasks. Additionally, both subsets include Visual Question Answering (VQA) question-answer pairs, further enhancing their utility for training and model evaluation.

In addition to introducing a novel dataset tailored for enhancing flowchart comprehension, this paper provides a rigorous analysis of the performance of contemporary LVLMs in interpreting flowcharts. Our findings reveal significant room for improvement in LVLMs, with no single model excelling across all tasks within the FlowLearn framework. For instance, in tasks involving simulated flowcharts, GPT-4V achieved the highest accuracy (58%) in counting the number of nodes, while Claude recorded the highest accuracy (83%) in OCR tasks. This varied performance highlights specific areas where LVLM capabilities could be further developed. Given the rapid advancements in the fields of Large Language Models (LLMs) and LVLMs, FlowLearn is both timely and valuable, providing a foundation for future research in visual data interpretation and automated reasoning, and setting new benchmarks in the field.

2 Related Works

This section overviews interdisciplinary research at the intersection of computer vision and natural language processing, focusing on the comprehension of visual figures in scientific contexts.

2.1 Scientific Figure Datasets

Significant efforts have been made to develop methodologies and datasets aimed at extracting and understanding scientific figures.

* Corresponding Author. Email: huitong.pan@temple.edu

[1] Our dataset is available on https://huggingface.co/datasets/jopan/FlowLearn. Our code for generation and evaluation is available on https://github.com/Jo-Pan/FlowLearn

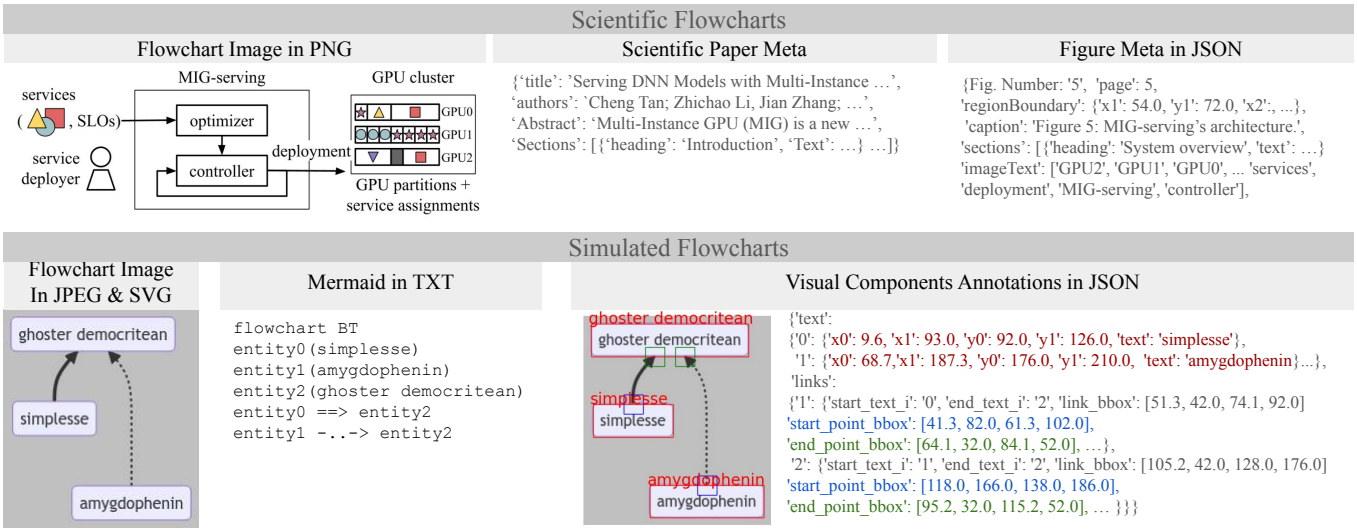


Figure 1: Overview of the FlowLearn Dataset illustrating the detailed components within the Scientific and Simulated subsets.

Methods such as PDFigCapX [16], PDFMEF [28], PDFFIGURES [6], and PDFFIGURES2 [5] facilitate the extraction of figures, captions, and related information from scholarly articles. Whereas datasets like VIS30K [4] and PDFFIGURES2 advance figure extraction by providing detailed annotations for figure locations. Moreover, ACL-Fig [13] and DocFigure [12] focus on figure classification, enhancing the understanding of various figure types, including bar charts and architecture diagrams. Additionally, specialized datasets like SciCap [11] and Parsing-AUC [24] concentrate on image captioning and summarization for experimental results figures.

Despite advancements, a significant gap exists in datasets with detailed annotations for flowcharts. For instance, ACL-Fig contains only about 208 flowcharts, primarily as architecture diagrams and neural networks, often duplicated, with brief captions or blurry figures from pre-2000 papers. CSDia [27] focuses on logical diagrams but lacks detailed caption information as it is not derived from the scientific literature.

2.2 *Visual Understanding Resources and Methods*

Research in image captioning [21, 34], particularly in scientific chart image captioning [8], has seen significant advancements, exemplified by works such as Parsing-AUC [24], which combines figure semantics extracted via OpenCV with textual information from the main text to generate comprehensive figure summaries for AUC figures.

The field of VQA [14, 1] has progressed substantially, with datasets like VL-ICL Bench [35] providing benchmarks for multimodal in-context learning. In scientific contexts, CSDia [27] employs models based on Diagram Parsing Nets to tackle VQA tasks. In scientific contexts, FigureSeer [26] automates figure localization, classification, and summarization. It extracts key visual components such as axes, legends, and data points for analysis, emphasizing the importance of figure decomposition in enhancing the understanding of scientific figures.

In non-scientific contexts, innovative approaches such as Neural-Symbolic VQA [29] and αILP [25] have been developed to transform neural network outputs into symbolic representations, which are then used to formulate answers. Additionally, the research highlighted in [15] stresses the importance of incorporating structural information in visualization retrieval processes. Their findings indicate a marked preference among survey participants for evaluating similarity based on visual elements rather than merely pixel-level details, suggesting a deeper, more structural approach to image analysis that could greatly benefit VQA systems.

Several works have focused on extracting objects and their relationships from scientific figures. Notably, [7] pioneers the conversion of scientific equation images into LaTeX format. ChartDetective [20] introduces an interactive application for converting result chart images into SVG, preserving semantics and component relationships. However, it is essential to note that this approach relies on user interaction and takes about 4 minutes for a single conversion. For non-scientific domains, Flow2Code [10] converts hand-drawn flowcharts to simple code by using object detection and rules.

The development of LVLMs has marked a significant leap in visual understanding, with models capable of integrating advanced vision techniques with LLMs. These models can learn simultaneously from images and texts, tackling various tasks such as visual question answering and image captioning. The OpenCompass Multimodal Leaderboard[2] rank these LVLMs, includes entries such as GPT-4V [33], Gemini [9], LLaVA [18], Claude [2], InternLM [31], Qwen-VL [3], Step-1V[3], and DeepSeek [19]. All of these models were trained on diverse datasets, including datasets for VQA, optical character recognition (OCR), and academic-related VQA. These are essential building blocks for achieving flowchart comprehension and enhancing the understanding of scientific flowcharts. However, among these models, only DeepSeek explicitly stated that its in-house training datasets included flowchart-related VQA. This highlights a potential area for future research and development, suggesting that incorporating more flowchart-specific data may enhance model training and performance.

In conclusion, there is a notable gap in resources specifically tailored for flowchart comprehension. Existing datasets and methods primarily focus on general scientific figures without addressing the unique complexities of flowcharts. Our FlowLearn dataset fills this gap by providing detailed annotations for flowcharts and evaluating LVLMs' ability to interpret these diagrams. By enhancing LVLMs' understanding of flowcharts, our work extends existing knowledge and introduces crucial resources for research aimed at enhancing automated visual reasoning and comprehension.

[2] https://rank.opencompass.org.cn/leaderboard-multimodal
[3] https://www.stepfun.com/

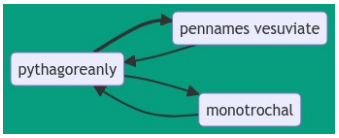
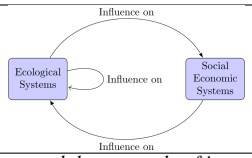
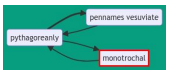
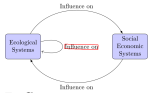
Task		Simulated Flowchart	Scientific Flowchart
			
OCR	Prompt	*A flowchart will be provided where a red box is drawn around the text node of interest. Answer with the text inside the red box. Ensure that the transcription is precise, reflecting the exact letters.*	
	Question		
	Answer	monotrochal	Influence on
True/False	Prompt	*The given image is a simulated flowchart. Based on the process outlined in the flowchart, determine the correctness of the given statement. Answer with either "true" or "false".*	*The given image is a flowchart extracted from a scientific literature. Based on the process outlined in the flowchart, determine the correctness of the given statement. Answer with either "true" or "false".*
	Question	There is an arrow between pennames vesuviate and monotrochal.	Ecological Systems can be influenced by itself.
	Answer	FALSE	TRUE
Description	Prompt	*The image contains a flowchart. Generate the description of the flowchart, reflecting the text nodes and arrows as depicted.*	
	Answer	pythagoreanly points to pennames vesuviate. pythagoreanly points to monotrochal. monotrochal points to pythagoreanly. pennames vesuviate points to pythagoreanly.	

Figure 7: Influence matrix schematic graph, based on [5, Figure 5]

Table 1: Common VQA tasks across both the Scientific and Simulated subsets of the FlowLearn Dataset.

Task		Simulated Flowchart
Mermaid Code	Prompt	*The image contains a flowchart. Generate the Mermaid code to represent the flowchart, reflecting the text nodes and arrows as depicted.*
	Answer	```mermaid flowchart LR entity0(pythagoreanly) entity1(monotrochal) entity2(pennames vesuviate) entity0 ==>entity2 entity0 –>entity1 entity1 –>entity0 entity2 –>entity0 ```
Num. of Nodes	Prompt	*The given image contains a simulated flowchart. You should find all text nodes and determine the total number of text nodes in the flowchart. Answer the question with a number.*
	Answer	3
Num. of Arrows	Prompt	*The given image contains a simulated flowchart. You should find all arrows and determine the total number of arrows in the flowchart. Answer the question with a number.*
	Answer	4

Table 2: VQA tasks unique to the Simulated Flowcharts subset of the FlowLearn Dataset.

3 FlowLearn Dataset

To address the scarcity of resources for flowchart comprehension, we introduce the FlowLearn Dataset. This dataset comprises two distinct subsets: Scientific Flowcharts and Simulated Flowcharts. An overview of the FlowLearn dataset, illustrating its components, is depicted in Figure 1. Table 1 details the common VQA tasks applicable to both subsets, while Table 2 lists the VQA tasks that are unique to the Simulated Flowcharts subset.

3.1 Scientific Flowchart Dataset

The Scientific Flowcharts Dataset comprises a large collection of flowchart images extracted from scholarly articles across diverse scientific domains. This dataset serves as a crucial resource for enhancing visual comprehension of scientific content.

We initiated our dataset creation by downloading 27,000 scientific articles from ArXiv. Using PDFFigures 2.0 [5], we extracted figures and related metadata. Additional metadata were parsed by the SciPDF Parser[4], which utilizes GROBID[5] for parsing PDFs.

Our selection process involved rule-based filtering combined with manual verification to identify flowcharts. We selected figures based on keywords relevant to flowcharts in captions, such as “illustration”, “flowchart”, “model”, “step”, “overall”, and “graphical representation”. Figures with unrelated keywords like "normalized" and "plot" were omitted. This meticulous curation yielded 3,858 flowcharts from 2,674 documents, focusing on images that prominently feature arrows, indicative of flowchart structures.

Each flowchart in our dataset is accompanied by comprehensive metadata, exemplified in Figure 1. The **Scientific Paper Meta** includes parsed text from the source articles, along with related information such as authors and titles. The **Figure Meta** encompasses the figure caption and the in-text reference of the figure. Additionally, we annotated all text appearing within each flowchart using PaddleOCR [30]. These annotations support various subtasks crucial to flowchart comprehension, including OCR and flowchart description.

3.2 Simulated Flowcharts

Recognizing that understanding flowcharts goes beyond caption generation, we developed the Simulated Flowcharts subset to enhance comprehension of diagrammatic components like arrows and nodes, which can be labor-intensive to annotate in scientific diagrams.

This subset was generated using Mermaid[6], a JavaScript tool that translates Markdown-inspired text definitions into flowcharts. Sample Mermaid code can be seen in Figure 1 and Table 2. We utilized Python scripts to introduce variability in the flowchart definitions in terms of the following aspects: **1) Nodes:** Each flowchart contains between 3 to 10 nodes, with node text consisting of randomized English words. **2) Links:** The number of links between nodes is randomized, with all nodes connected by at least one link, mimicking real-world flowchart structures. We randomize the type of arrow links between nodes, including solid lines, bold lines, or dashed lines. **3) Background Color and Flowchart Orientation:** Background col-

[4] https://github.com/titipata/scipdf_parser
[5] https://github.com/kermitt2/grobid
[6] https://mermaid.js.org/

ors are randomly generated in hexadecimal format. The orientation of the flowcharts is randomized, encompassing all available options in the Mermaid syntax.

We generated a total of 10,000 samples. Each sample includes: **1) Flowchart Images:** Available in JPEG and SVG formats. **2)Mermaid Code:** Provided for each sample to facilitate programmatic understanding and manipulation of the flowchart structure. **3)Visual Component Annotations:** Detailed annotations are provided, which include the node text and the precise locations of text nodes, arrowheads, and tails, all derived from SVG. These annotations are crucial for tasks such as object detection and structural analysis, enabling a deeper understanding of the flowchart components.

The generation script provides fine-grained control over the creation of simulated flowchart samples, enabling integrated training and experimentation for a wide range of applications.

3.3 Visual Question Answering

To evaluate the flowchart understanding capabilities using the FlowLearn dataset, we developed tailored VQA question-answer pairs for each tested flowchart. Examples of prompts, questions and answers for each task are detailed in Table 1 and Table 2. We have ensured that all prompts are elaborately detailed based on findings from VL-ICL [35], which demonstrated that more detailed prompts significantly enhance VQA performance compared to shorter ones. Our own experiments confirm this finding, as we observed that detailed prompts consistently outperform shorter ones in eliciting accurate responses from models.

The common VQA tasks for both subsets include:

OCR: We randomly place a red box over one of the annotated texts within the flowchart and prompt models to identify and return the enclosed words.

True/False: We generate statements related to the flowchart and query the model to determine their veracity. For *Scientific Flowcharts*, we initially create two accurate statements using sentences from the figure caption, subsequently verified by annotators for their correctness based on the flowchart. In cases with insufficient caption data, annotators generate additional statements relate to the flowchart. For false statements, annotators alter a few words in a true statement to reverse its meaning, ensuring the vocabulary remains consistent with the original author's style. This process yields one true and one false statement for each tested scientific flowchart.

For *Simulated Flowcharts*, we use predefined templates to create True and False statements, such as: "An arrow exists between node '{a}' and node '{b}'" and "An arrow points from node '{a}' to node '{b}'." where {a} and {b} are placeholders for node texts identified in Visual Component Annotations (Section 3.2).

Description: We prompt models to generate descriptions for the flowcharts. For *scientific flowcharts*, the reference answers are derived from their captions; for *simulated flowcharts*, reference answers are generated by converting mermaid code to sentences using templates, "{a} points to {b}."

Additionally, the Simulated Flowcharts includes 3 unique tasks:

Mermaid Code: Models are tasked with generating Mermaid code that represents the flowchart. This task assesses the model's ability to comprehensively recognize flowchart components, including text nodes and arrows.

Number of Nodes and Arrows: Models answer questions regarding the count of text nodes and arrows present in the flowchart. This task offers a quantitative measure of the model's comprehension, though it is less comprehensive than the Mermaid Code task.

4 Experiment Setups

In this section, we detail the experimental setup used to assess the capabilities of various Large Vision-Language Models (LVLMs) using the FlowLearn Dataset. Our primary objective is to evaluate how effectively these models comprehend and interpret flowcharts from both the Scientific and Simulated subsets. We have implemented all VQA tasks outlined in Section 3.3, which probe various facets of flowchart comprehension—from fundamental text recognition to more comprehensive overall understanding.

4.1 Models

We selected LVLMs for evaluation based on their rankings in the OpenCampass multi-modal leaderboard as of April 2024. Access to some models was facilitated through APIs, including Step-1V-32K, GPT-4V, Gemini-Pro-Vision, and Claude-3-Opus-20240229. Additional models assessed in our study were LLaVA-V1.6-Vicuna-34B, InternLM-XComposer2-VL-7B, Qwen-VL-Chat from 2024/01/25, and DeepSeek-VL-7B-chat. Our selection strategy aimed to choose the best model available from each top-ranked model family, such as selecting Claude-3-Opus from the Claude series.

4.2 Evaluation Metrics

To evaluate the performance of the models, we categorized the VQA tasks into three groups, each assessed by tailored evaluation metrics:

Accuracy: We measure the accuracy for tasks including OCR, True/False Statements, Number of Nodes, and Number of Arrows. This metric is straightforward and evaluates whether the responses are correct or incorrect based on the ground truth. Specifically for True/False Statements, we calculated average accuracy separately for the true and false subsets, and an overall average accuracy to provide a comprehensive view of model performance.

Similarity: For description tasks, we assess the closeness of model-generated descriptions to reference descriptions using four similarity metrics: BLEU [22], ROUGE-L [17], BERT score [32] and Sentence Transformer Similarity (SBERT) [23]. The BERT score utilizes pre-trained BERT embeddings to assess semantic coherence through cosine similarity of matched words. Similarly, SBERT converts both response and reference sentences into embeddings with the 'all-MiniLM-L6-v2' model, using cosine similarity to quantitatively gauge how closely the generated text matches the target description. We also calculate median word count of responses.

Mermaid Code Generation: We developed two sets of metrics specifically tailored for evaluating the correctness of generated Mermaid code:

- *Node-Level Evaluation:* This metric checks if all nodes present in the ground truth are included in the model's response. Each node is only considered correct if it exactly matches the spelling in the ground truth.
- *Link-Level Evaluation:* This metric assesses the generated response includes all the links present in the ground truth. A link is deemed correct if both the start and end nodes are accurately predicted, regardless of the arrow type. We also permit some syntactical flexibility in how node descriptions are expressed, allowing the use of either the node variable name or the node text.

For both evaluation levels, we compute F1-score, precision, and recall for each sample and average these metrics across all samples.

4.3 Response Parsing

Given the variability in how LVLMs generate responses, which may not always exactly match the ground truth even when correct, we have developed specific rules to parse and evaluate the responses:

OCR: A prediction is deemed correct if it includes the exact phrase from the ground truth.

True/False Statements: The response is assessed for the presence of 'true' or 'false', case-insensitively. Responses lacking these tokens or containing both are marked as incorrect.

Number of Nodes or Arrows: We extract the first numeric token in the response, also converting English words representing numbers into numeric tokens. If no such token appears, the response is marked as incorrect.

Mermaid Code Prediction: We focus on statements encapsulated within triple backticks (```) in model responses. From these, we extract nodes and links according to the Mermaid syntax rules.

4.4 Settings

For our evaluations, we utilized the testing subset of the FlowLearn dataset, which included assessments of 500 scientific flowcharts and 2,000 simulated flowcharts. Due to cost constraints and API limitations, we limited our evaluations to 100 samples per task for Claude-3-Opus, GPT-4V, and Step-1V. All other evaluations were conducted using an NVIDIA A100 80GB GPU.

We opted for few-shot prompting as our evaluation strategy to align the output of the LVLMs more closely with the ground truth. According to Zong et al. [35], few-shot prompting, particularly with 2-shot samples, generally yields the most significant performance improvement in general vision-language VQA tasks across various LVLMs. Additionally, using 2-shot samples provides a balanced approach for evaluating True/False statements, as it allows an equal representation of both true and false scenarios within the prompts. This method ensures that the models are not biased toward one answer type over the other, facilitating a more accurate and fair assessment of model capabilities.

For consistency, we employ the prompt format shown in Table 3 for evaluation.

Prompt: [Task Description] **Support Set:** [Image][Question][Answer] (2-shot) **Query:** [Image][Question] **Prediction:** [Answer]

Table 3: 2-Shot prompt format used for evaluation.

5 Experiment Results

In this section, we present the results from our evaluation of the LVLMs across three distinct groups of VQA tasks within the FlowLearn dataset. Each task group was designed to test different aspects of model performance using specialized evaluation metrics. For a focused review of performance across a limited subset of 100 samples involving all models and all tasks, please refer to Section 2 of the Supplementary Materials. The findings there align closely with the results discussed here. Sample model responses to all VQA tasks are shown in Section 3 and 4 of the Supplementary Materials.

5.1 Accuracy Tasks

The first group of tasks evaluates the accuracy of the LVLMs in responding to queries that require precise, binary, or short phrase answers. These tasks are foundational for assessing flowchart comprehension. The performance of each model on these accuracy tasks is summarized in Table 4, leading to several key observations:

1) **No clear winner across all accuracy tasks.** For scientific flowcharts, Gemini-Pro-Vision showed the strongest performance on the full test set. However, on smaller subsets, GPT-4V and Step-1V also demonstrated strong performances. For simulated flowcharts, on the full test set, InternLM excelled in True/False statements, Gemini-Pro in OCR tasks, and Qwen-VL in counting nodes and arrows.

2) **Irrelevant model responses.** Although most models generally produced task-related responses, irrelevant responses were still observed. For True/False tasks, Qwen-VL and LLaVA often scored close to zero, indicating a lack of 'true' or 'false' tokens in its responses.

3) **Challenges in counting nodes and arrows.** Counting tasks, which require comprehensive image understanding rather than partial recognition, proved difficult for most models, leading to lower average scores. Notably, despite its underperformance in other areas, Qwen-VL's results were comparatively better in these tasks.

5.2 Similarity Tasks (Description)

The second group of tasks assesses LVLMs' ability to generate accurate descriptions of flowcharts, with detailed performance data for each model presented in Table 5.

For scientific flowcharts, DeepSeek and Gemini achieved the highest scores on most metrics. Qwen-VL produced the longest responses, while DeepSeek provided the shortest. We also reviewed responses from the widely-used GPT4V. Out of 100 samples, only 24 were error-free. Analysis at the sentence level yielded 597 sentences with a 59% accuracy rate, meaning 41% of sentences contained errors. Sentences were marked as incorrect only if evidence from the image contradicted the text. Supplementary material includes examples. We identified several common trends:

1) Generally, models provided satisfactory responses, accurately inferring full names from acronyms and offering reasonable academic interpretations of the depicted processes.

2) Models handled complex images effectively, including those with high resolutions and multiple parts.

3) Longer descriptions were more error-prone, whereas shorter ones, while accurate, lacked comprehensive coverage, explaining the lower sample-wise compared to sentence-wise accuracy.

4) In many cases, models produced logical but inaccurate descriptions. Examples of this are shown in Table 6.

For simulated flowcharts, Gemini outperformed other models across most metrics, typically scoring higher than for scientific flowcharts. This discrepancy likely stems from the structured, template-generated reference answers for simulated flowcharts, contrasted with the varied language and additional contextual information in scientific flowchart captions.

5.3 Mermaid Code Task

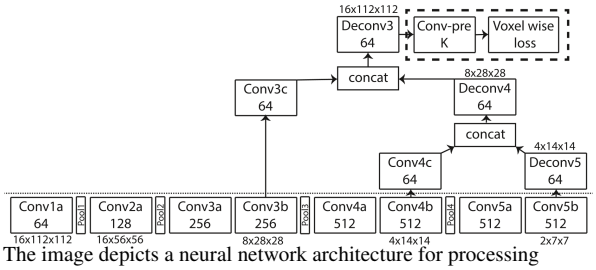
This task assesses the comprehensive ability of LVLMs to encapsulate their understanding of a flowchart in a code format, summarizing aspects such as OCR, counting nodes and arrows, and recognizing relationships between nodes. The performance of each model on the Mermaid Code task for simulated flowcharts is summarized in Table 7. In evaluations on the full dataset, Gemini achieved the highest scores across all metrics. On a smaller evaluation subset, Claude demonstrated superior performance, particularly excelling in node-level prediction with an F1 score of 94%.

Task		Claude†	GPT4V†	Step-1V†	Gemini ProVision	InternLM -XComposer2-VL	LLaVA 16-34B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Scientific Flowchart									
Statements	OCR	0.44	0.51	**0.66**	0.43	0.06	0.01	0.08	0.05
	TRUE	0.69	0.63	**0.9**	0.55	0.89	0.15	0.18	0.86
	FALSE	0.53	**0.74**	0.36	0.7	0.28	0	0	0.21
	Average	0.61	**0.68**	0.63	0.62	0.59	0.08	0.09	0.53
Simulated Flowchart									
	OCR	**0.83**	0.75	0.71	0.69	0.18	0	0.58	0.23
	Num. Nodes	0.52	**0.58**	0.31	0.02	0.15	0.12	0.4	0.22
	Num. Arrows	0.23	**0.26**	**0.26**	0.09	0.12	0.1	0.2	0.15
Statements: Between AB	TRUE	0.42	0.28	0.62	0.69	**0.82**	0.05	0.34	0.61
	FALSE	0.71	0.77	0.72	0.52	0.5	0.01	0	0.56
	Average	0.56	0.52	**0.67**	0.61	0.66	0.03	0.17	0.59
Statements: A to B	TRUE	0.18	0.15	0.41	0.16	**0.85**	0.11	0.04	0.55
	FALSE	0.49	0.61	0.5	0.63	0.49	0.03	0.01	0.58
	Average	0.34	0.38	0.45	0.4	**0.67**	0.07	0.03	0.57

Table 4: Experiment results for accuracy tasks. Models† are evaluated on a subset of the evaluation set. Regardless of evaluation size, the best-performing model is **bolded**. The best-performing model among those evaluated on the full set is underlined.

Evaluation Metrics	Claude†	GPT4V†	Step-1V†	Gemini ProVision	InternLM -XComposer2-VL	LLaVA 16-34B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Scientific Flowchart								
Median Word Count	98	97	65	190	101	19	354	9
BLEU	0.47	0.43	0.33	**0.49**	0.05	0	0.02	0.13
ROUGE-L	0.13	0.13	0.12	0.09	0.1	0.11	0.07	**0.17**
BERTScore-F1	0.84	0.83	0.84	0.83	0.83	0.81	0.79	**0.86**
Sent.Transformer Similarity	**0.49**	0.46	0.42	0.30	0.34	0.25	0.38	0.36
Simulated Flowchart								
Median Word Count	28	34	35	33	58	19	200	32
BLEU	**0.02**	0.01	0.01	0.01	0.01	0	0	**0.02**
ROUGE-L	0.54	0.52	0.52	**0.56**	0.19	0	0.13	0.41
BERTScore-F1	0.90	0.90	0.89	**0.92**	0.82	0.77	0.80	0.87
Sent.Transformer Similarity	0.84	0.84	0.84	**0.88**	0.41	0.18	0.51	0.71

Table 5: Experiment results for Flowchart Description task. Models† are evaluated on a subset of the evaluation set. Regardless of evaluation size, the best-performing model is **bolded**. The best-performing model among those evaluated on the full set is underlined.



The image depicts a neural network architecture for processing volumetric data, possibly for a segmentation task. It features an encoder-decoder structure with skip connections. Here is the flow:
1. The network inputs a volume of 16x112x112x1.
2. It goes through four convolutional layers with the following specifications:
- Conv1a: 64 feature maps, output size 16x112x112
- Conv2a: 128 feature maps, output size 16x56x56
- Conv3a: 256 feature maps, output size 8x28x28
- Conv4a: 512 feature maps, output size 4x14x14
3. There's another convolutional layer Conv5a with 512 feature maps reducing the volume size to 2x7x7.
... *(cropped)* ...
Lastly, there's a Conv-pre layer (size not specified) followed by Bounding Box (BB) Regression to predict 'K' classes, and a voxel-wise loss is computed at the final output.
Note: The sizes of the feature maps in the decoder part are not explicitly stated, but they are likely to match the corresponding encoder layer sizes due to skip connections. The dotted lines indicate feedback of loss or gradients back through the network during training.

Table 6: Example GPT-4V's response (cropped) to the Flowchart-to-Caption tasks on Scientific Flowcharts from the FlowLearn Dataset. Incorrect parts are highlighted in red.

Challenges were notable in models like InternLM, LLaVA, Qwen-VL, and DeepSeek, all of which recorded scores close to zero. Several issues were identified during the evaluation of their outputs:

Syntax Compliance: These models did not adhere to the proper syntax of Mermaid code, failing to correct their outputs even after 2-shot prompting designed to teach them the correct code format.

Node Recognition: Disregarding syntax issues, these models still struggled to accurately predict correct nodes. The node-level evaluation, which also indirectly assesses models' OCR capabilities by checking for the presence of all node text in the predictions, reflected poor performance. This aligns with results from Table 4, where these models underperformed in OCR tasks that required text detection within specified areas.

Link-level predictions, which depend on accurate node-level results, consider a prediction correct only if the start and end nodes and the direction of the link are identified accurately. Hence, scores for link-level evaluations generally fall below those for node-level evaluations. Claude, which scored highly at the node level, encountered many challenges with link prediction, achieving only a 30% F1 score for link-level accuracy. This highlights the difficulty models face in understanding complex relationships within flowcharts.

5.4 *Ablation Study on Chain-of-Thought*

For complex tasks such as converting a flowchart into Mermaid code, a methodical approach can be beneficial. This process typically involves several sequential steps: initially detecting text nodes, then recognizing the links between them, and finally compiling these information into a standardized format, such as Mermaid code. Given the multi-step nature of this task, we hypothesized that introducing

Evaluation Metric		Claude†	GPT4V†	Step-1V†	Gemini ProVision	Gemini ProVision (CoT)	InternLM -XComposer2-VL	LLaVA 16-34B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Link	Precision	**0.35**	0.23	0.14	0.26	0.25	0.01	0	0.02	0.05
	Recall	**0.26**	0.22	0.15	0.25	0.24	0.02	0	0.02	0.04
	F1	**0.3**	0.22	0.14	0.25	0.25	0.02	0	0.02	0.04
Node	Precision	**0.94**	0.72	0.68	0.75	0.71	0.09	0	0.06	0.29
	Recall	**0.95**	0.73	0.68	0.75	0.71	0.16	0	0.08	0.29
	F1	**0.94**	0.72	0.68	0.75	0.71	0.12	0	0.07	0.28

Table 7: Results for Flowchart-to-Mermaid on Simulated Flowcharts. Models† are evaluated on a subset of the evaluation set. Regardless of evaluation size, the best-performing model is **bolded**. The best-performing model among those evaluated on the full set is underlined.

a chain-of-thought (CoT) process could potentially enhance model performance. Consequently, we conducted an experimental ablation study on the simulated subset using Gemini-Pro-Vision, a model that incurs no querying cost and can be evaluated on the full test set. Notably, this model has shown the best performance on the Mermaid code task (Section 5.3).

First, the flowchart includes the following nodes: **{1}**
Then, it contains the following edges: **{2}**
Finally, the Mermaid code for the flowchart is: **{3}**

Table 8: Chain-of-Thought answer template.

For this experiment, we modified the 2-shot example answers using a structured template (Table 8) that guides the model through a step-by-step reasoning process. In this template, {1} is replaced with all text appearing in the simulated flowchart, {2} is derived from the flowchart description generated as per the templates described in Section 3.3, and {3} is the corresponding Mermaid code. Additionally, we appended the phrase "Let's think step by step" at the end of the original prompt (as illustrated in Table 2) to further emphasize the sequential reasoning process.

We selected Gemini, the top-performing model for the Mermaid Code Task, for this ablation study. Surprisingly, as shown in Table 7, the performance using the CoT approach indicated a slight decrease compared to the original model configuration without it. This unexpected outcome suggests that while the chain-of-thought method is intended to foster clearer and more structured reasoning, it may introduce additional complexities or dependencies that hinder the model's ability to synthesize and process information efficiently. Further analysis and refinement of the chain-of-thought implementation may be necessary to fully realize its potential benefits and address these challenges.

6 Discussion

The initial version of the FlowLearn dataset presents inherent limitations, offering opportunities for future enhancements.

6.1 Scientific Flowchart Subset

First, True/False statements are missing for the training set of scientific flowcharts. Annotators generate these statements only for test samples, which is time-consuming, averaging three minutes per pair. Future versions should aim to include these statements for all entries.

Second, the dataset size is currently limited. With fewer than 4,000 images, FlowLearn's scientific flowchart collection is small compared to larger visual-language datasets. While the inclusion of simulated flowcharts helps to mitigate this limitation by broadening the scope of the training data, expanding the collection of scientific flowcharts would be advantageous.

Third, the descriptive task is limited. The descriptive task for scientific flowcharts is currently evaluated against figure captions. However, the descriptive text for scientific diagrams is often scattered throughout the associated literature, as outlined in Context24[7]: Contextualizing Scientific Figures and Tables. A more robust approach would involve annotators extracting and collating descriptive text from the full text of scientific articles to provide a more comprehensive base for evaluating LVLM-generated descriptions.

6.2 Simulated Flowchart Subset

The simulated flowchart subset was designed to augment the scientific subset by offering a more granular evaluation of flowchart comprehension and providing additional training data. Future iterations could improve upon this by incorporating a greater diversity of diagram types, such as state diagrams and quadrant charts, to enrich the dataset further. While FlowLearn currently focuses exclusively on flowcharts, expanding the range of diagram types could enhance its applicability.

6.3 Model Selection

Our model selection was biased towards LVLMs due to their broad capabilities and general applicability. However, many task-specific, smaller visual-language models may also be well-suited for these tasks. Future work will explore the potential of these models, which might offer more specialized insights or efficiencies in specific aspects of flowchart comprehension.

7 Conclusion

In this study, we introduced and evaluated the FlowLearn dataset, a novel resource aimed at advancing the understanding of flowcharts for visual-language models. Our experiments spanned various tasks, including OCR, True/False assessments, counting nodes and arrows, flowchart description, and generating Mermaid code, across two distinct subsets: scientific and simulated flowcharts.

Our findings demonstrate that while LVLMs are capable of impressive performance on certain tasks, challenges remain. Notably, the models excelled at OCR and True/False statements in certain contexts but struggled with the more complex task of accurately generating Mermaid code from flowcharts. This underscores a broader issue: LVLMs often struggle to fully comprehend the intricate relationships between visual components and to synthesize this information into structured code formats effectively.

Given the rapid advancements in the fields of LLMs and LVLMs, the FlowLearn dataset is timely and provides valuable insights into a relatively underexplored area. It not only serves as a critical tool for benchmarking and refining these models but also helps illuminate the specific difficulties they encounter with visual reasoning in a structured context. By pushing the boundaries of what LVLMs can understand and achieve, we can bridge the gap between human and machine comprehension of visual and language tasks, paving the way for more intelligent and capable automated systems.

[7] https://sdproc.org/2024/sharedtasks.html#context24

Acknowledgements

This work was supported by the National Science Foundation awards III-2107213, III-2107518, and ITE-2333789. We also thank undergraduate students Eric Reizas and Elle Nguyen at Temple University for their valuable contributions to our project.

References

[1] A. Z. Al-Jamal, M. J. Bani-Amer, and S. A. Aljawarneh. Image captioning techniques: A review. *2022 International Conference on Engineering & MIS (ICEMIS)*, pages 1–5, 2022. URL https://api.semanticscholar.org/CorpusID:252900543.

[2] Anthropic. The claude 3 model family: Opus, sonnet, haiku. URL https://api.semanticscholar.org/CorpusID:268232499.

[3] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond, 2023.

[4] J. Chen, M. Ling, R. Li, P. Isenberg, T. Isenberg, M. Sedlmair, T. Möller, R. S. Laramee, H.-W. Shen, K. Wünsche, and Q. Wang. Vis30k: A collection of figures and tables from ieee visualization conference publications. *IEEE Transactions on Visualization and Computer Graphics*, 27 (9):3826–3833, 2021. doi: 10.1109/TVCG.2021.3054916.

[5] C. Clark and S. Divvala. Pdffigures 2.0: Mining figures from research papers. 2016.

[6] C. Clark and S. K. Divvala. Looking beyond text: Extracting figures, tables and captions from computer science papers. In *AAAI Workshop: Scholarly Big Data*, 2015. URL https://api.semanticscholar.org/CorpusID:16285667.

[7] Y. Deng, A. Kanervisto, J. Ling, and A. M. Rush. Image-to-markup generation with coarse-to-fine attention. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 980–989. JMLR.org, 2017.

[8] A. M. Farahani, P. Adibi, M. S. Ehsani, H.-P. Hutter, and A. Darvishy. Automatic chart understanding: A review. *IEEE Access*, 11:76202–76221, 2023. doi: 10.1109/ACCESS.2023.3298050.

[9] C. Fu, R. Zhang, Z. Wang, Y. Huang, Z. Zhang, L. Qiu, G. Ye, Y. Shen, M. Zhang, P. Chen, S. Zhao, S. Lin, D. Jiang, D. Yin, P. Gao, K. Li, H. Li, and X. Sun. A challenger to gpt-4v? early explorations of gemini in visual expertise, 2023.

[10] J.-I. Herrera-Camara and T. Hammond. Flow2code: from hand-drawn flowcharts to code execution. In *Proceedings of the Symposium on Sketch-Based Interfaces and Modeling*, SBIM '17, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350792. doi: 10.1145/3092907.3092909. URL https://doi.org/10.1145/3092907.3092909.

[11] T.-Y. Hsu, C. L. Giles, and T.-H. Huang. SciCap: Generating captions for scientific figures. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3258–3264, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.277. URL https://aclanthology.org/2021.findings-emnlp.277.

[12] K. V. Jobin, A. Mondal, and C. V. Jawahar. Docfigure: A dataset for scientific document figure classification. *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, Sep 2019. doi: https://doi.org/10.1109/icdarw.2019.00018. URL https://ieeexplore-ieee-org.libproxy.temple.edu/document/8892905.

[13] Z. Karishma, S. Rohatgi, K. S. Puranik, J. Wu, and C. L. Giles. Acl-fig: A dataset for scientific figure classification, 2023.

[14] V. Kodali and D. Berleant. Recent, rapid advancement in visual question answering: a review. *2022 IEEE International Conference on Electro Information Technology (eIT)*, pages 139–146, 2022. URL https://api.semanticscholar.org/CorpusID:250367882.

[15] H. Li, Y. Wang, A. Wu, H. Wei, and H. Qu. Structure-aware visualization retrieval. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3502048. URL https://doi.org/10.1145/3491102.3502048.

[16] P. Li, X. Jiang, and H. Shatkay. Extracting figures and captions from scientific publications. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1595–1598, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450360142. doi: 10.1145/3269206.3269265. URL https://doi.org/10.1145/3269206.3269265.

[17] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Annual Meeting of the Association for Computational Linguistics*, 2004. URL https://api.semanticscholar.org/CorpusID:964287.

[18] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning, 2023.

[19] H. Lu, W. Liu, B. Zhang, B. Wang, K. Dong, B. Liu, J. Sun, T. Ren, Z. Li, H. Yang, Y. Sun, C. Deng, H. Xu, Z. Xie, and C. Ruan. Deepseek-vl: Towards real-world vision-language understanding, 2024.

[20] D. Masson, S. Malacria, D. Vogel, E. Lank, and G. Casiez. Chartdetective: Easy and accurate interactive data extraction from complex vector charts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450394215. doi: 10.1145/3544548.3581113. URL https://doi.org/10.1145/3544548.3581113.

[21] Y. Ming, N. Hu, C. Fan, F. Feng, J. Zhou, and H. Yu. Visuals to text: A comprehensive review on automatic image captioning. *IEEE/CAA Journal of Automatica Sinica*, 9(8):1339–1365, 2022. doi: 10.1109/JAS.2022.105734.

[22] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics*, 2002. URL https://api.semanticscholar.org/CorpusID:11080756.

[23] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*, 2019. URL https://api.semanticscholar.org/CorpusID:201646309.

[24] I. Safder, H. Batool, R. Sarwar, F. Zaman, N. R. Aljohani, R. Nawaz, M. Gaber, and S.-U. Hassan. Parsing auc result-figures in machine learning specific scholarly documents for semantically-enriched summarization. *Applied Artificial Intelligence*, 36(1):2004347, 2022. doi: 10.1080/08839514.2021.2004347. URL https://doi.org/10.1080/08839514.2021.2004347.

[25] H. Shindo, V. Pfanschilling, D. S. Dhami, and K. Kersting. alphailp: thinking visual scenes as differentiable logic programs. *Machine Learning*, 112(5):1465–1497, Mar 2023. doi: https://doi.org/10.1007/s10994-023-06320-1. URL https://link.springer.com/article/10.1007/s10994-023-06320-1#citeas.

[26] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi. Figureseer: Parsing result-figures in research papers. In *European Conference on Computer Vision (ECCV)*, 2016.

[27] S. Wang, L. Zhang, X. Luo, Y. Yang, X. Hu, T. Qin, and J. Liu. Computer science diagram understanding with topology parsing. *ACM Trans. Knowl. Discov. Data*, 16(6), jul 2022. ISSN 1556-4681. doi: 10.1145/3522689. URL https://doi.org/10.1145/3522689.

[28] J. Wu, J. Killian, H. Yang, K. Williams, S. R. Choudhury, S. Tuarob, C. Caragea, and C. L. Giles. Pdfmef: A multi-entity knowledge extraction framework for scholarly documents and semantic search. In *Proceedings of the 8th International Conference on Knowledge Capture*, K-CAP 2015, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450338493. doi: 10.1145/2815833.2815834. URL https://doi.org/10.1145/2815833.2815834.

[29] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In *Neural Information Processing Systems*, 2018. URL https://api.semanticscholar.org/CorpusID:52919654.

[30] D. Yu, X. Li, C. Zhang, T. Liu, J. Han, J. Liu, and E. Ding. Towards accurate scene text recognition with semantic reasoning networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12113–12122, 2020.

[31] P. Zhang, X. Dong, B. Wang, Y. Cao, C. Xu, L. Ouyang, Z. Zhao, H. Duan, S. Zhang, S. Ding, W. Zhang, H. Yan, X. Zhang, W. Li, J. Li, K. Chen, C. He, X. Zhang, Y. Qiao, D. Lin, and J. Wang. Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition, 2023.

[32] T. Zhang*, V. Kishore*, F. Wu*, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

[33] X. Zhang, Y. Lu, W. Wang, A. Yan, J. Yan, L. Qin, H. Wang, X. Yan, W. Y. Wang, and L. R. Petzold. Gpt-4v(ision) as a generalist evaluator for vision-language tasks, 2023.

[34] Z. Zohourianshahzadi and J. K. Kalita. Neural attention for image captioning: review of outstanding methods. *Artificial Intelligence Review*, 55:3833 – 3862, 2021. URL https://api.semanticscholar.org/CorpusID:244729197.

[35] Y. Zong, O. Bohdal, and T. Hospedales. Vl-icl bench: The devil in the details of benchmarking multimodal in-context learning, 2024.

8 Supplementary Materials

8.1 Preliminary experiment with ACL-Fig

We conducted a preliminary experiment using flowcharts from ACL-Fig, where flowcharts are categorized as neural networks and architecture diagrams. The dataset includes a total of 208 flowcharts. We queried Gemini-Pro-Vision with a prompt requesting a figure description, using the same prompt shown in the main text. By comparing the generated descriptions to the actual figure captions, we obtained surprisingly poor results: BLEU score of 0.006, ROUGE-L of 0.08, BERTScore-F1 of 0.824, and Sentence Transformer Similarity of 0.40. We also manually evaluated the generated descriptions and found that almost all responses contained at least one error, such as incorrect interpretation of an arrow leading to erroneous causation explanations. Examples of these descriptions are illustrated in Table 9.

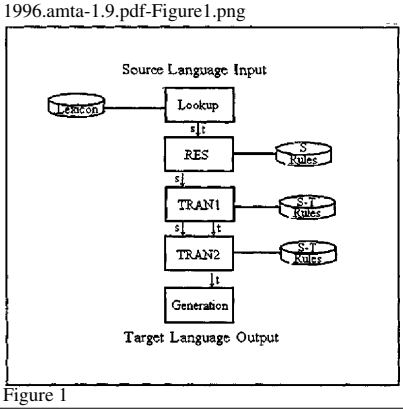
Image File	1996.amta-1.9.pdf-Figure1.png
Image	
Caption	Figure 1
Description from Gemini	The flowchart describes the process of translating a source language into a target language. The process starts with the source language input, which is then looked up in a lexicon. The result of the lookup is then passed to a series of modules, which perform various transformations on the input. The first module is the RES module, which is responsible for resolving anaphora. The output of the RES module is then passed to the S-Rules module, which applies a set of rules to the input. The output of the S-Rules module is then passed to the TRANI module, which performs a series of transformations on the input. The output of the TRANI module is then passed to the TRAN2 module, which performs a series of transformations on the input. The output of the TRAN2 module is then passed to the Generation module, which generates the target language output.

Table 9: Example description from Gemini-Pro-Vision for a sample in ACL-Fig. Text highlighted in red indicates parts that incorrectly describe the figure.

8.2 Experiment Results on a Limited Subset

Due to cost and API limitations, the models Claude, GPT-4V, and Step-1V were evaluated on a smaller subset of 100 samples. To provide a comprehensive comparison, we present the experiment results for all models across this smaller subset in Table 10, Table 11, and Table 12. The findings from this limited subset are consistent with the results discussed in the main text, underscoring the robustness and reliability of our earlier observations.

8.3 Sample Model Responses for all tasks

In this section, we showcase a selection of response examples from various models. These examples demonstrate how different models processed and responded to different Visual Question Answering (VQA) tasks across both scientific and simulated flowcharts within the FlowLearn dataset. The responses are displayed in Table 13, Table 14, and Table 15. For each VQA task, we aim to include at least one effective response and one less effective response to highlight the range of model capabilities and limitations.

8.4 GPT4V for Flowchart-to-Caption

We conducted a sentence-level evaluation of GPT-4V's responses in the Flowchart-to-Caption task. We divided the 100 responses into 597 sentences. A PhD student specializing in NLP research was tasked with evaluating the responses. The table presents sample findings as discussed in the main text.

	Task	Claude	GPT4V	Step-1V	Gemini ProVision	InternLM -XComposer2-VL	LLaVA16-34B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Scientific Flowchart									
	OCR	0.44	0.51	**0.66**	0.53	0.05	0.01	0.07	0.09
Statements	TRUE	0.69	0.63	**0.9**	0.63	0.84	0.14	0.17	0.82
	FALSE	0.53	**0.74**	0.36	0.61	0.32	0	0	0.21
	Average	0.61	**0.68**	0.63	0.62	0.58	0.07	0.08	0.52
Simulated Flowchart									
	OCR	**0.83**	0.75	0.71	0.64	0.19	0	0.6	0.2
	Num. Nodes	0.52	**0.58**	0.31	0.04	0.15	0.08	0.42	0.22
	Num. Arrows	0.23	**0.26**	**0.26**	0.07	0.07	0.07	0.19	0.12
Statements: Between AB	TRUE	0.42	0.28	0.62	0.68	**0.79**	0.26	0.34	0.74
	FALSE	0.71	**0.77**	0.72	0.49	0.44	0.01	0	0.5
	Average	0.56	0.52	**0.67**	0.58	0.62	0.14	0.17	0.62
Statements: A to B	TRUE	0.18	0.15	0.41	0.13	**0.83**	0.79	0.04	0.51
	FALSE	0.49	**0.61**	0.5	0.53	0.4	0.33	0.02	0.52
	Average	0.34	0.38	0.45	0.33	**0.62**	0.56	0.03	0.52

Table 10: Experiment results for accuracy tasks across the same 100 samples. The best-performing model is highlighted in **bolded**.

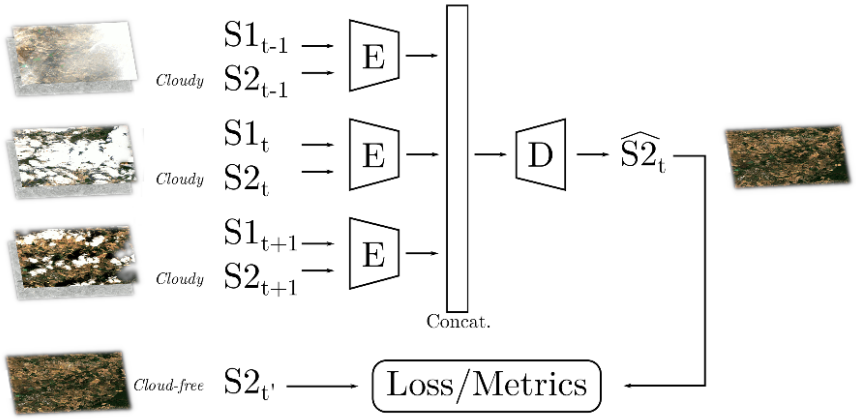
Evaluation Metrics	Claude	GPT4V	Step-1V	Gemini ProVision	InternLM -XComposer2-VL	LLaVA16-7B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Scientific Flowchart								
BLEU	0.02	0.01	0.01	0.01	0.01	0	0	**0.03**
ROUGE-L	0.13	0.13	0.12	0.1	0.1	0.11	0.07	**0.18**
BERTScore-F1	0.84	0.83	0.83	0.83	0.83	0.84	0.8	**0.86**
Sent.Transformer Similarity	**0.49**	0.46	0.39	0.3	0.33	0.19	0.38	0.36
Simulated Flowchart								
BLEU	0.47	0.43	0.33	**0.48**	0.04	0	0.02	0.11
ROUGE-L	0.54	0.52	0.52	**0.56**	0.18	0	0.12	0.4
BERTScore-F1	0.9	0.9	0.89	**0.91**	0.82	0.79	0.79	0.86
Sent.Transformer Similarity	0.84	0.84	0.84	**0.87**	0.41	0.06	0.5	0.69

Table 11: Experiment results for Flowchart Description task across the same 100 samples. The best-performing model is highlighted in **bolded**.

Evaluation Metrics		Claude	GPT4V	Step-1V	Gemini ProVision	Gemini ProVision(CoT)	InternLM -XComposer2-VL	LLaVA16-34B	Qwen-VL -chat	DeepSeek -VL-7B-chat
Link	Precision	**0.35**	0.23	0.14	0.21	0.23	0.01	0	0.01	0.02
	Recall	**0.26**	0.22	0.15	0.2	0.22	0.01	0	0.01	0.02
	F1	**0.3**	0.22	0.14	0.21	0.22	0.01	0	0.01	0.02
Node	Precision	**0.94**	0.72	0.68	0.7	0.68	0.09	0	0.05	0.24
	Recall	**0.95**	0.73	0.68	0.71	0.68	0.16	0	0.06	0.24
	F1	**0.94**	0.72	0.68	0.7	0.68	0.11	0	0.05	0.24

Table 12: Experiment results for Flowchart-to-Mermaid on Simulalted Flowcharts across the same 100 samples. The best-performing model is highlighted in **bolded**.

Dataset	FlowLearn - Scientific Flowcharts
Example File	2204.00424v1-Figure6-1.png



Task	Flowchart-to-Caption
Reference	Figure 6. MSOPunet network. (S1t-1, S2t-1), (S1t, S2t) and (S1t+1, S2t+1) denotes the input pairs of optical and SAR images. S2t' denotes the reference image used to compute the loss (during training) and the metrics (at inference time from test data) from the reconstructed optical image S2t.
GPT4V	The answer is a flowchart depicting a process for handling satellite images to predict cloud-free images. The steps involve the following: 1. Input of a pair of images at time t-1 (S1_t-1 and S2_t-1), labeled as "Cloudy." 2. They are each passed through an encoder (E) to produce encoded representations. 3. Another pair of images at time t with labels "Cloudy" for S1_t and "Cloud-free" for S2_t are also passed through the same encoder (E). 4. Another image at time t+1 (S2_t+1) labeled as "Cloudy" is encoded similarly. 5. All the encoded representations are concatenated. 6. The encoded concatenation is passed through a decoder (D) to generate an output (S2_t). 7. Finally, this output is compared to a target image and assessed using Loss/Metrics to evaluate the performance of the prediction.
InternLM-XComposer2-VL	The image is a flowchart that shows the architecture of BERT in pretraining. The model's task is to predict the masked tokens from the representation vectors it learns at the positions of the input embeddings. The diagram also includes a section on Loss/Metrics, which indicates how the loss is calculated and what metrics are used to evaluate the performance of the model during training.
Task	OCR
Reference	Loss/Metrics
GeminiProVision	Loss/Metrics
Qwen-VL-chat	1 Picture 4: 2 The loss metrics are computed using
LLaVA16-34B	C1
Task	Statement: True
Question Statement	The output of 3 encoder layers are concatenated.
Reference	true
GPT4V	false
Claude	true
Task	Statement: False
Question Statement	Cloudy image is used to calculate loss.
Reference	false
GeminiProVision	false
DeepSeek-VL-7B-chat	true

Table 13: Models' responses to all VQA tasks on a Scientific Flowchart from the FlowLearn Dataset. Correct responses are highlighted in teal, and incorrect responses are marked in red.

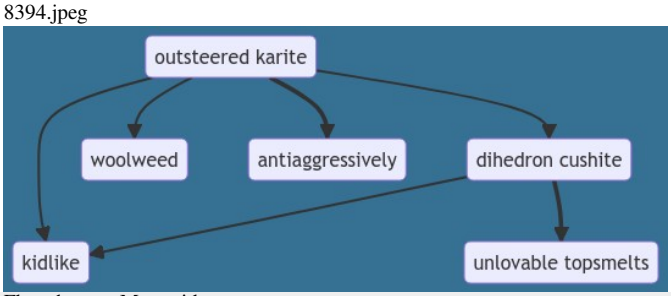
Dataset	SimFlowchart-word
Example File	8394.jpeg
	
Task	Flowchart-to-Mermaid
Reference	‘‘‘mermaid flowchart TB entity0(outsteered karite) entity1(dihedron cushite) entity2(woolweed) entity3(kidlike) entity4(antiaggressively) entity5(unlovable topsmelts) entity0 –>entity1 entity0 –>entity2 entity0 –>entity3 entity0 ==>entity4 entity1 –>entity3 entity1 ==>entity5 ‘‘‘
GeminiProVision	‘‘‘mermaid flowchart LR entity0(outsteered karite) entity1(woolweed) entity2(antiaggressively) entity3(dihedron cushtie) entity4(kidlike) entity5(unlovable topsmelts) entity0 –>entity1 entity0 –>entity2 entity0 –>entity3 entity1 –>entity4 entity3 –>entity5 ‘‘‘
Qwen-VL-chat (cropped)	1. outsteered karite 2. woolweed 3. antiaggresively 4. dihedron cushite 5. kidlike 6. unlovable topsmelts 7. karite 8. woolweed 9. antiaggresively 10. dihedron cushite 11. kidlike 12. unlovable topsmelts 13. karite 14. woolweed 15. antiaggresively 16. dihedron cushite 17. kidlike 18. unlovable topsmelts 19. karite 20. woolweed 21. antiaggresively 22. dihedron cushite 23. kidlike 24. unlovable topsmelts 25. karite 26. woolweed 27. antiaggresively ...

Table 14: Models' responses to the first segment of VQA tasks on an evaluated Simulated Flowchart from the FlowLearn Dataset. Correct responses are highlighted in teal, and incorrect responses are marked in red.

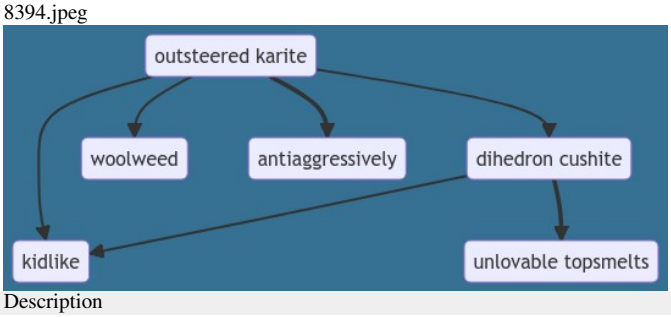
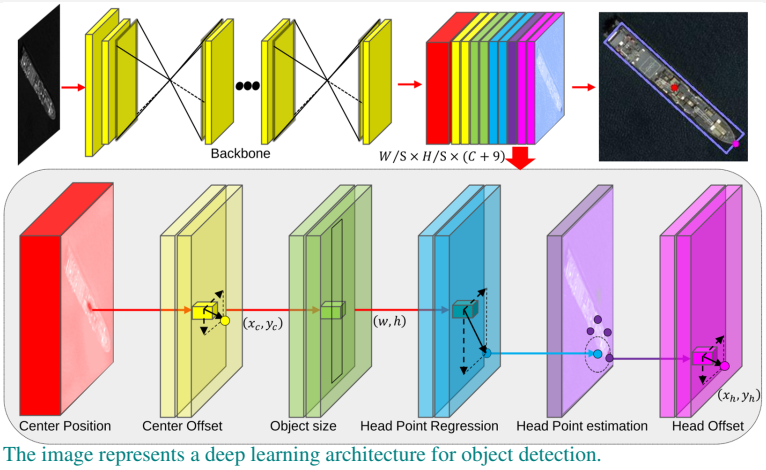
Dataset	SimFlowchart-word
Example File	8394.jpeg
	
Task	Description
Reference	outsteered karite points to dihedron cushite. outsteered karite points to woolweed. outsteered karite points to kidlike. outsteered karite points to antiaggressively. dihedron cushite points to kidlike. dihedron cushite points to unlovable topsmelts.
LLaVA16-34B	The image contains a flowchart. Generate the description of the flowchart, reflecting the text nodes and arrows as depicted.
Claude	The answer is: outsteered karite points to woolweed. outsteered karite points to antiaggresively. woolweed points to kidlike. antiaggresively points to dihedron cushite. dihedron cushite points to unlovable topsmelts.
Task	OCR
Reference	outsteered karite
InternLM-XComposer2-VL	kidlike
DeepSeek-VL-7B-chat	lansing
Step-1V	outsteered karite
Task	Num_Nodes
Reference	6
GPT4V	6
GeminiProVision	The total number of text nodes in the flowcharts is 13.
Task	Num_Arrows
Reference	6
Claude	7
Step-1V	5
Task	AtoB Statement: True
Question Statement	Arrow points from node 'dihedron cushite' to node 'unlovable topsmelts'.
Reference	true
InternLM-XComposer2-VL	true
GPT4V	false
Task	AtoB Statement: False
Question Statement	Arrow points from node 'kidlike' to node 'outsteered karite'.
Reference	false
DeepSeek-VL-7B-chat	false
InternLM-XComposer2-VL	true
Task	Between AB Statement: False
Question Statement	Arrow exists between node 'dihedron cushite' and node 'kidlike'.
Reference	false
GPT4V	false
DeepSeek-VL-7B-chat	true
Task	Between AB Statement: True
Question Statement	Arrow exists between node 'dihedron cushite' and node 'woolweed'.
Reference	true
Claude	false
Step-1V	true

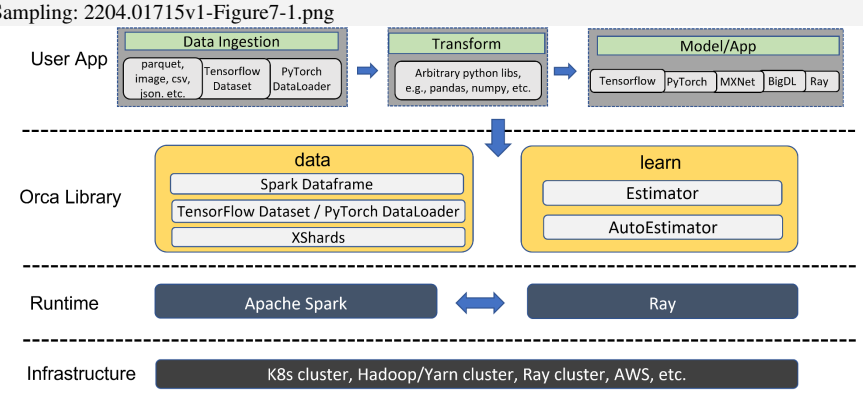
Table 15: Models' responses to the second segment of VQA tasks on an evaluated Simulated Flowchart from the FlowLearn Dataset. Correct responses are highlighted in teal, and incorrect responses are marked in red.

Related Finding: 1) Satisfactory responses 2)complex images 3) long-response
Sample: 2101.11189v1-Figure2-1.png



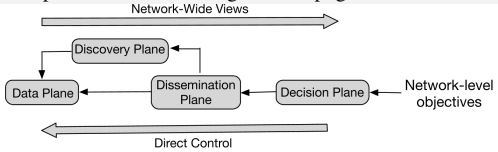
The image represents a deep learning architecture for object detection.
It consists of the following sequence of key components:
1.Backbone:A neural network used for initial feature extraction from the input
2.A series of feature maps of different spatial resolutions, denoted by W/S × H/S × (C + 9).
This likely represents feature maps with width and height scaled by a factor of S, containing C channels plus an additional 9 channels.
3. These feature maps are processed through different branches for detecting various attributes of objects:
- Center Position: Predicts the center (x_c, y_c) of an object.
- Center Offset: Compensates for errors in the predicted object center.
- Object size: Estimates the width (w) and height (h) of the object.
- Head Point Regression and Head Point Estimate: Localize a specific point on the object, usually the "head" point.
- Head Offset: Adjusts for errors in the head point prediction.

Related Finding: 2)complex images 3) long-response 4) Logical but inaccurate descriptions, such as incorrectly interpreting the direction of arrows.
Sampling: 2204.01715v1-Figure7-1.png



The answer is Figure 8: An overview of the system architecture.
The system is structured in three main layers: Infrastructure, Runtime, and User App.
In the Infrastructure layer, there are options such as K8s cluster, Hadoop/Yarn cluster, Ray cluster, AWS, etc.
The Runtime layer consists of Apache Spark.
In the User App layer, the flow starts from "Data Ingestion", moving to "Transform", and then to "Model/App".
For "Data Ingestion" some mentioned inputs are image, csv, json, etc., and tools like Parquet, TensorFlow Dataset, or PyTorch Dataset can be used.
The "Transform" step includes tools such as Spark DataFrame, TensorFlow Dataset/PyTorch DataLoader, and XShards.
For the "Model/App", examples include PyTorch, BigDL, Ray, etc.
The "data" component from the User App flows into the Orca Library in the Runtime layer, which then proceeds to the "learn"
component with Estimator and AutoEstimator.
The information moves back and forth as indicated by the bidirectional arrows between "data" and "learn".

Sample: 2011.04105v1-Figure3.5-1.png



The flowchart describes the structure of network views, with three primary planes and their interactions.
Starting at the "Data Plane," there are two arrows branching out; one leads to the "Discovery Plane" and another flows directly to the "Dissemination Plane."
From the Discovery Plane, the flow moves to the Dissemination Plane.
Then, from the Dissemination Plane, the process progresses to the "Decision
The Decision Plane then aligns with "Network-level objectives.
"There are also two broader categories depicted as horizontal arrows spanning the entire process: "Network-Wide Views" at the top and "Direct Control" at the bottom, indicating overarching concepts that encompass the specific planes.

Table 16: GPT-4V's responses to the Flowchart-to-Caption tasks on Scientific Flowcharts from the FlowLearn Dataset, evaluated at the sentence level. Sentences are highlighted to indicate accuracy: correct sentences in teal and incorrect sentences in red.